BSN: Boundary Sensitive Network for Temporal Action Proposal Generation

Tianwei Lin[1], Xu Zhao*[1], Haisheng Su[1], Chongjing Wang[2], Ming Yang[1]

[1]Department of Automation, Shanghai Jiao Tong University, China
[2]China Academy of Information and Communications Technology, China
wzmsltw,zhaoxu,suhaisheng,mingyang@sjtu.edu.cn
wangchongjing@caict.ac.cn

Abstract. Temporal action proposal generation is an important yet challenging problem, since temporal proposals with rich action content are indispensable for analysing real-world videos with long duration and high proportion irrelevant content. This problem requires methods not only generating proposals with precise temporal boundaries, but also retrieving proposals to cover truth action instances with high recall and high overlap using relatively fewer proposals. To address these difficulties, we introduce an effective proposal generation method, named Boundary-Sensitive Network (BSN), which adopts ***"local to global"*** fashion. ***Locally***, BSN first locates temporal boundaries with high probabilities, then directly combines these boundaries as proposals. ***Globally***, with Boundary-Sensitive Proposal feature, BSN retrieves proposals by evaluating the confidence of whether a proposal contains an action within its region. We conduct experiments on two challenging datasets: ActivityNet-1.3 and THUMOS14, where BSN outperforms other state-of-the-art temporal action proposal generation methods with high recall and high temporal precision. Finally, further experiments demonstrate that by combining existing action classifiers, our method significantly improves the state-of-the-art temporal action detection performance.

Keywords: Temporal action proposal generation · Temporal action detection · Temporal convolution · Untrimmed video

1 Introduction

Nowadays, with fast development of digital cameras and Internet, the number of videos is continuously booming, making automatic video content analysis methods widely required. One major branch of video analysis is action recognition, which aims to classify manually trimmed video clips containing only one action instance. However, videos in real scenarios are usually long, untrimmed and contain multiple action instances along with irrelevant contents. This problem requires algorithms for another challenging task: temporal action detection, which aims to detect action instances in untrimmed video including both temporal boundaries and action classes. It can be applied in many areas such as video recommendation and smart surveillance.

Similar with object detection in spatial domain, temporal action detection task can be divided into two stages: proposal and classification. Proposal generation stage aims

* Corresponding author.

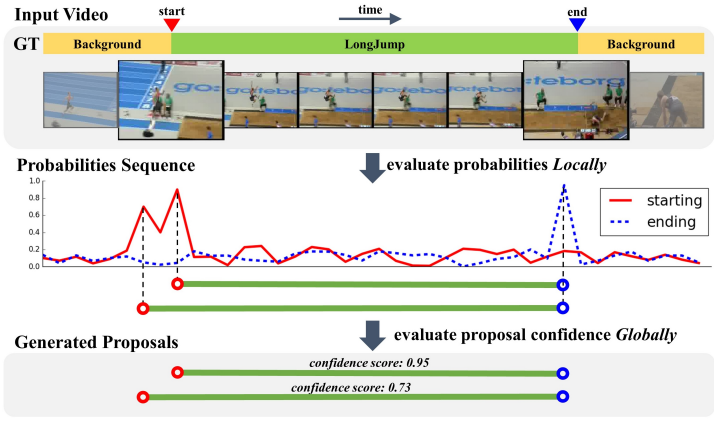


Fig. 1: Overview of our approach. Given an untrimmed video, (1) we evaluate boundaries and actionness probabilities of each temporal location and generate proposals based on boundary probabilities, and (2) we evaluate the confidence scores of proposals with proposal-level feature to get retrieved proposals.

to generate temporal video regions which may contain action instances, and classification stage aims to classify classes of candidate proposals. Although classification methods have reached convincing performance, the detection precision is still low in many benchmarks [1,2]. Thus recently temporal action proposal generation has received much attention [3,4,5,6], aiming to improve the detection performance by improving the quality of proposals. High quality proposals should come up with two key properties: (1) proposals can cover truth action regions with both high recall and high temporal overlap, (2) proposals are retrieved so that high recall and high overlap can be achieved using fewer proposals to reduce the computation cost of succeeding steps.

To achieve high proposal quality, a proposal generation method should generate proposals with flexible temporal durations and precise temporal boundaries, then retrieve proposals with reliable confidence scores, which indicate the probability of a proposal containing an action instance. Most recently proposal generation methods [3,4,5,7] generate proposals via sliding temporal windows of multiple durations in video with regular interval, then train a model to evaluate the confidence scores of generated proposals for proposals retrieving, while there is also method [6] making external boundaries regression. However, proposals generated with pre-defined durations and intervals may have some major drawbacks: (1) usually not temporally precise; (2) not flexible enough to cover variable temporal durations of ground truth action instances, especially when the range of temporal durations is large.

To address these issues and generate high quality proposals, we propose the Boundary-Sensitive Network (BSN), which adopts "*local to global*" fashion to locally combine high probability boundaries as proposals and globally retrieve candidate proposals using proposal-level feature as shown in Fig 1. In detail, BSN generates proposals in three steps. **First**, BSN evaluates the probabilities of each temporal location in video whether it is inside or outside, at or not at the boundaries of ground truth action instances, to generate starting, ending and actionness probabilities sequences as local information.

Second, BSN generates proposals via directly combining temporal locations with high starting and ending probabilities separately. Using this bottom-up fashion, BSN can generate proposals with flexible durations and precise boundaries. **Finally**, using features composed by actionness scores within and around proposal, BSN retrieves proposals by evaluating the confidence of whether a proposal contains an action. These proposal-level features offer global information for better evaluation.

In summary, the main contributions of our work are three-folds:

(1) We introduce a new architecture (BSN) based on "*local to global*" fashion to generate high quality temporal action proposals, which *locally* locates high boundary probability locations to achieve precise proposal boundaries and *globally* evaluates proposal-level feature to achieve reliable proposal confidence scores for retrieving.

(2) Extensive experiments demonstrate that our method achieves significantly better proposal quality than other state-of-the-art proposal generation methods, and can generate proposals in unseen action classes with comparative quality.

(3) Integrating our method with existing action classifier into detection framework leads to significantly improved performance on temporal action detection task.

2 Related work

Action recognition. Action recognition is an important branch of video related research areas and has been extensively studied. Earlier methods such as improved Dense Trajectory (iDT) [8,9] mainly adopt hand-crafted features such as HOF, HOG and MBH. In recent years, convolutional networks are widely adopted in many works [10,11,12,13] and have achieved great performance. Typically, two-stream network [10,11,13] learns appearance and motion features based on RGB frame and optical flow field separately. C3D network [12] adopts 3D convolutional layers to directly capture both appearance and motion features from raw frames volume. Action recognition models can be used for extracting frame or snippet level visual features in long and untrimmed videos.

Object detection and proposals. Recent years, the performance of object detection has been significantly improved with deep learning methods. R-CNN [14] and its variations [15,16] construct an important branch of object detection methods, which adopt "detection by classifying proposals" framework. For proposal generation stage, besides sliding windows [17], earlier works also attempt to generate proposals by exploiting low-level cues such as HOG and Canny edge [18,19]. Recently some methods [16,20,21] adopt deep learning model to generate proposals with faster speed and stronger modelling capacity. In this work, we combine the properties of these methods via evaluating boundaries and actionness probabilities of each location using neural network and adopting "*local to global*" fashion to generate proposals with high recall and accuracy.

Boundary probabilities are also adopted in LocNet [22] for revising the horizontal and vertical boundaries of existing proposals. Our method differs in (1) BSN aims to generate while LocNet aims to revise proposals and (2) boundary probabilities are calculated repeatedly for all boxes in LocNet but only once for a video in BSN.

Temporal action detection and proposals. Temporal action detection task aims to detect action instances in untrimmed videos including temporal boundaries and action classes, and can be divided into proposal and classification stages. Most detection methods [7,23,24] take these two stages separately, while there is also method [25,26] taking

these two stages jointly. For proposal generation, earlier works [27,28,29] directly use sliding windows as proposals. Recently some methods [3,4,5,6,7] generate proposals with pre-defined temporal durations and intervals, and use multiple methods to evaluate the confidence score of proposals, such as dictionary learning [4] and recurrent neural network [5]. TAG method [24] adopts watershed algorithm to generate proposals with flexible boundaries and durations in *local* fashion, but without *global* proposal-level confidence evaluation for retrieving. In our work, BSN can generate proposals with flexible boundaries meanwhile reliable confidence scores for retrieving.

Recently temporal action detection method [30] detects action instances based on class-wise start, middle and end probabilities of each location. Our method is superior than [30] in two aspects: (1) BSN evaluates probabilities score using temporal convolution to better capture temporal information and (2) "*local to global*" fashion adopted in BSN brings more precise boundaries and better retrieving quality.

3 Our Approach

3.1 Problem Definition

An untrimmed video sequence can be denoted as $X = \{x_n\}_{n=1}^{l_v}$ with l_v frames, where x_n is the n-th frame in X. Annotation of video X is composed by a set of action instances $\Psi_g = \{\varphi_n = (t_{s,n}, t_{e,n})\}_{n=1}^{N_g}$, where N_g is the number of truth action instances in video X, and $t_{s,n}$, $t_{e,n}$ are starting and ending time of action instance φ_n separately. Unlike detection task, classes of action instances are not considered in temporal action proposal generation. Annotation set Ψ_g is used during training. During prediction, generated proposals set Ψ_p should cover Ψ_g with high recall and high temporal overlap.

3.2 Video Features Encoding

To generate proposals of input video, first we need to extract feature to encode visual content of video. In our framework, we adopt two-stream network [11] as visual encoder, since this architecture has shown great performance in action recognition task [31] and has been widely adopted in temporal action detection and proposal generation tasks [24,25,32]. Two-stream network contains two branches: spatial network operates on single RGB frame to capture appearance feature, and temporal network operates on stacked optical flow field to capture motion information.

To extract two-stream features, as shown in Fig 2(a), first we compose a snippets sequence $S = \{s_n\}_{n=1}^{l_s}$ from video X, where l_s is the length of snippets sequence. A snippet $s_n = (x_{t_n}, o_{t_n})$ includes two parts: x_{t_n} is the t_n-th RGB frame in X and o_{t_n} is stacked optical flow field derived around center frame x_{t_n}. To reduce the computation cost, we extract snippets with a regular frame interval σ, therefore $l_s = l_v/\sigma$. Given a snippet s_n, we concatenate output scores in top layer of both spatial and temporal networks to form the encoded feature vector $f_{t_n} = (f_{S,t_n}, f_{T,t_n})$, where f_{S,t_n}, f_{T,t_n} are output scores from spatial and temporal networks separately. Thus given a snippets sequence S with length l_s, we can extract a feature sequence $F = \{f_{t_n}\}_{n=1}^{l_s}$. These two-stream feature sequences are used as the input of BSN.

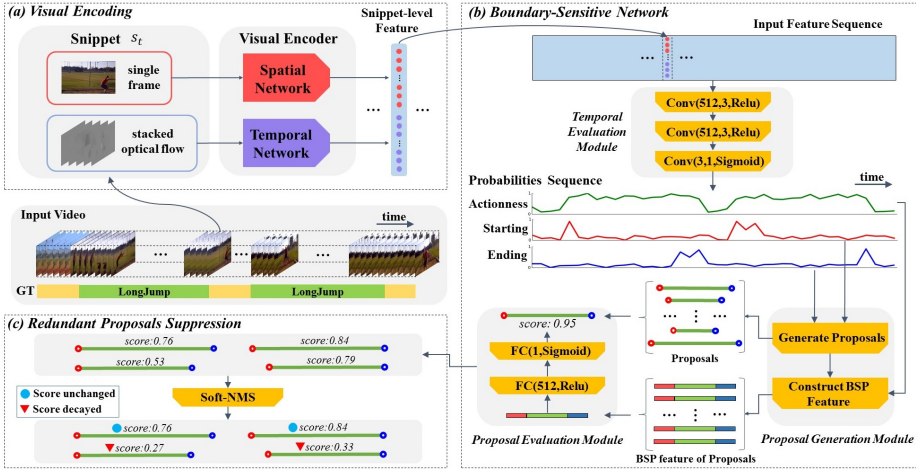


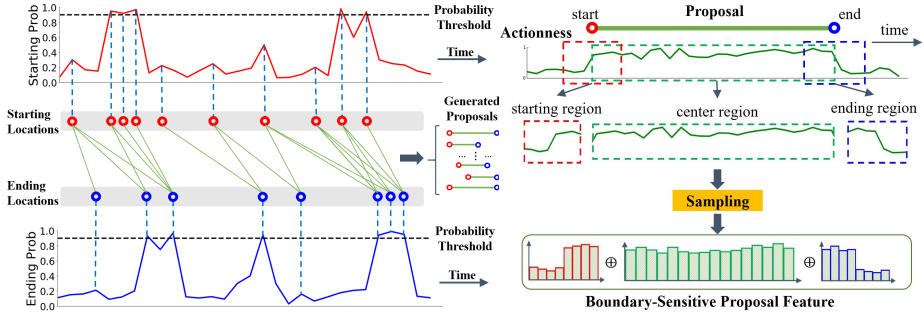
Fig. 2: The framework of our approach. (a) Two-stream network is used for encoding visual features in snippet-level. (b) The architecture of Boundary-Sensitive Network: *temporal evaluation module* handles the input feature sequence, and evaluates starting, ending and actionness probabilities of each temporal location; *proposal generation module* generates proposals with high starting and ending probabilities, and construct Boundary-Sensitive Proposal (BSP) feature for each proposal; *proposal evaluation module* evaluates confidence score of each proposal using BSP feature. (c) Finally, we use Soft-NMS algorithm to suppress redundant proposals by decaying their scores.

3.3 Boundary-Sensitive Network

To achieve high proposal quality with both precise temporal boundaries and reliable confidence scores, we adopt "*local to global*" fashion to generate proposals. In BSN, we first generate candidate boundary locations, then combine these locations as proposals and evaluate confidence score of each proposal with proposal-level feature.

Network architecture. The architecture of BSN is presented in Fig 2(b), which contains three modules: temporal evaluation, proposal generation and proposal evaluation. *Temporal evaluation module* is a three layers temporal convolutional neural network, which takes the two-stream feature sequences as input, and evaluates probabilities of each temporal location in video whether it is inside or outside, at or not at boundaries of ground truth action instances, to generate sequences of starting, ending and actionness probabilities respectively. *Proposal generation module* first combines the temporal locations with separately high starting and ending probabilities as candidate proposals, then constructs Boundary-Sensitive Proposal (BSP) feature for each candidate proposal based on actionness probabilities sequence. Finally, *proposal evaluation module*, a multilayer perceptron model with one hidden layer, evaluates the confidence score of each candidate proposal based on BSP feature. Confidence score and boundary probabilities of each proposal are fused as the final confidence score for retrieving.

Temporal evaluation module. The goal of temporal evaluation module is to evaluate starting, ending and actionness probabilities of each temporal location, where three binary classifiers are needed. In this module, we adopt temporal convolutional layers upon



(a) Generate proposals (b) Construct BSP feature

Fig. 3: Details of proposal generation module. (a) Generate proposals. First, to generate candidate boundary locations, we choose temporal locations with high boundary probability or being a probability peak. Then, we combine candidate starting and ending locations as proposals when their duration satisfying condition. (b) Construct BSP feature. Given a proposal and actionness probabilities sequence, we can sample actionness sequence in starting, center and ending regions of proposal to construct BSP feature.

feature sequence, with good modelling capacity to capture local semantic information such as boundaries and actionness probabilities.

A temporal convolutional layer can be simply denoted as $Conv(c_f, c_k, Act)$, where c_f, c_k and Act are filter numbers, kernel size and activation function of temporal convolutional layer separately. As shown in Fig 2(b), the temporal evaluation module can be defined as $Conv(512, 3, Relu) \rightarrow Conv(512, 3, Relu) \rightarrow Conv(3, 1, Sigmoid)$, where the three layers have same stride size 1. Three filters with sigmoid activation in the last layer are used as classifiers to generate starting, ending and actionness probabilities separately. For convenience of computation, we divide feature sequence into non-overlapped windows as the input of temporal evaluation module. Given a feature sequence F, temporal evaluation module can generate three probability sequences $P_S = \{p^s_{t_n}\}_{n=1}^{l_s}$, $P_E = \{p^e_{t_n}\}_{n=1}^{l_s}$ and $P_A = \{p^a_{t_n}\}_{n=1}^{l_s}$, where $p^s_{t_n}$, $p^e_{t_n}$ and $p^a_{t_n}$ are respectively starting, ending and actionness probabilities in time t_n.

Proposal generation module. The goal of proposal generation module is to generate candidate proposals and construct corresponding proposal-level feature. We achieve this goal in two steps. First we locate temporal locations with high boundary probabilities, and combine these locations to form proposals. Then for each proposal, we construct Boundary-Sensitive Proposal (BSP) feature.

As shown in Fig 3(a), to locate where an action likely to start, for starting probabilities sequence P_S, we record all temporal location t_n where $p^s_{t_n}$ (1) has high score: $p^s_{t_n} > 0.9$ or (2) is a probability peak: $p^s_{t_n} > p^s_{t_{n-1}}$ and $p^s_{t_n} > p^s_{t_{n+1}}$. These locations are grouped into candidate starting locations set $B_S = \{t_{s,i}\}_{i=1}^{N_S}$, where N_S is the number of candidate starting locations. Using same rules, we can generate candidate ending locations set B_E from ending probabilities sequence P_E. Then, we generate temporal regions via combing each starting location t_s from B_S and each ending location t_e from B_S. Any temporal region $[t_s, t_e]$ satisfying $d = t_e - t_s \in [d_{min}, d_{max}]$ is denoted as a candidate proposal φ, where d_{min} and d_{max} are minimum and maximum durations

of ground truth action instances in dataset. Thus we can get candidate proposals set $\Psi_p = \{\varphi_i\}_{i=1}^{N_p}$, where N_p is the number of proposals.

To construct proposal-level feature as shown in Fig 3(b), for a candidate proposal φ, we denote its center region as $r_C = [t_s, t_e]$ and its starting and ending region as $r_S = [t_s - d/5, t_s + d/5]$ and $r_E = [t_e - d/5, t_e + d/5]$ separately. Then, we sample the actionness sequence P_A within r_c as f_c^A by linear interpolation with 16 points. In starting and ending regions, we also sample actionness sequence with 8 linear interpolation points and get f_s^A and f_e^A separately. Concatenating these vectors, we can get Boundary-Sensitive Proposal (BSP) feature $f_{BSP} = (f_s^A, f_c^A, f_e^A)$ of proposal φ. BSP feature is highly compact and contains rich semantic information about corresponding proposal. Then we can represent a proposal as $\varphi = (t_s, t_e, f_{BSP})$.

Proposal evaluation module. The goal of proposal evaluation module is to evaluate the confidence score of each proposal whether it contains an action instance within its duration using BSP feature. We adopt a simple multilayer perceptron model with one hidden layer as shown in Fig 2(b). Hidden layer with 512 units handles the input of BSP feature f_{BSP} with Relu activation. The output layer outputs confidence score p_{conf} with sigmoid activation, which estimates the overlap extent between candidate proposal and ground truth action instances. Thus, a generated proposal can be denoted as $\varphi = (t_s, t_e, p_{conf}, p_{t_s}^s, p_{t_e}^e)$, where $p_{t_s}^s$ and $p_{t_e}^e$ are starting and ending probabilities in t_s and t_e separately. These scores are fused to generate final score during prediction.

3.4 Training of BSN

In BSN, temporal evaluation module is trained to learn local boundary and actionness probabilities from video features simultaneously. Then based on probabilities sequence generated by trained temporal evaluation module, we can generate proposals and corresponding BSP features and train the proposal evaluation module to learn the confidence score of proposals. The training details are introduced in this section.

Temporal evaluation module. Given a video X, we compose a snippets sequence S with length l_s and extract feature sequence F from it. Then we slide windows with length $l_w = 100$ in feature sequence without overlap. A window is denoted as $\omega = \{F_\omega, \Psi_\omega\}$, where F_ω and Ψ_ω are feature sequence and annotations within the window separately. For ground truth action instance $\varphi_g = (t_s, t_e)$ in Ψ_ω, we denote its region as action region r_g^a and its starting and ending region as $r_g^s = [t_s - d_g/10, t_s + d_g/10]$ and $r_g^e = [t_e - d_g/10, t_e + d_g/10]$ separately, where $d_g = t_e - t_s$.

Taking F_ω as input, temporal evaluation module generates probabilities sequence $P_{S,\omega}$, $P_{E,\omega}$ and $P_{A,\omega}$ with same length l_w. For each temporal location t_n within F_ω, we denote its region as $r_{t_n} = [t_n - d_s/2, t_n + d_s/2]$ and get corresponding probability scores $p_{t_n}^s$, $p_{t_n}^e$ and $p_{t_n}^a$ from $P_{S,\omega}$, $P_{E,\omega}$ and $P_{A,\omega}$ separately, where $d_s = t_n - t_{n-1}$ is temporal interval between two snippets. Then for each r_{t_n}, we calculate its IoP ratio with r_g^a, r_g^s and r_g^e of all φ_g in Ψ_ω separately, where IoP is defined as the overlap ratio with groundtruth proportional to the duration of this proposal. Thus we can represent information of t_n as $\phi_n = (p_{t_n}^a, p_{t_n}^s, p_{t_n}^e, g_{t_n}^a, g_{t_n}^s, g_{t_n}^e)$, where $g_{t_n}^a$, $g_{t_n}^s$, $g_{t_n}^e$ are maximum matching overlap IoP of action, starting and ending regions separately.

Given a window of matching information as $\Phi_\omega = \{\phi_n\}_{n=1}^{l_s}$, we can define training objective of this module as a three-task loss function. The overall loss function consists of actionness loss, starting loss and ending loss:

$$L_{TEM} = \lambda \cdot L_{bl}^{action} + L_{bl}^{start} + L_{bl}^{end}, \tag{1}$$

where λ is the weight term and is set to 2 in BSN. We adopt the sum of binary logistic regression loss function L_{bl} for all three tasks, which can be denoted as:

$$L_{bl} = \frac{1}{l_w} \sum_{i=1}^{l_w} \left(\alpha^+ \cdot b_i \cdot log(p_i) + \alpha^- \cdot (1 - b_i) \cdot log(1 - p_i)\right), \tag{2}$$

where $b_i = sign(g_i - \theta_{IoP})$ is a two-values function for converting matching score g_i to $\{0, 1\}$ based on threshold θ_{IoP}, which is set to 0.5 in BSN. Let $l^+ = \sum g_i$ and $l^- = l_w - l^+$, we can set $\alpha^+ = \frac{l_w}{l^-}$ and $\alpha^- = \frac{l_w}{l^+}$, which are used for balancing the effect of positive and negative samples during training.

Proposal evaluation module. Using probabilities sequences generated by trained temporal evaluation module, we can generate proposals using proposal generation module: $\Psi_p = \{\varphi_n = (t_s, t_e, f_{BSP})\}_{n=1}^{N_p}$. Taking f_{BSP} as input, for a proposal φ, confidence score p_{conf} is generated by proposal evaluation module. Then we calculate its Intersection-over-Union (IoU) with all φ_g in Ψ_g, and denote the maximum overlap score as g_{iou}. Thus we can represent proposals set as $\Psi_p = \{\varphi_n = \{t_s, t_e, p_{conf}, g_{iou}\}\}_{n=1}^{N_p}$. We split Ψ_p into two parts based on g_{iou}: Ψ_p^{pos} for $g_{iou} > 0.7$ and Ψ_p^{neg} for $g_{iou} < 0.3$. For data balancing, we take all proposals in Ψ_p^{pos} and randomly sample the proposals in Ψ_p^{neg} to insure the ratio between two sets be nearly 1:2.

The training objective of this module is a simple regression loss, which is used to train a precise confidence score prediction based on IoU overlap. We can define it as:

$$L_{PEM} = \frac{1}{N_{train}} \sum_{i=1}^{N_{train}} (p_{conf,i} - g_{iou,i})^2, \tag{3}$$

where N_{train} is the number of proposals used for training.

3.5 Prediction and Post-processing

During prediction, we use BSN with same procedures described in training to generate proposals set $\Psi_p = \left\{\varphi_n = (t_s, t_e, p_{conf}, p_{t_s}^s, p_{t_e}^e)\right\}_{n=1}^{N_p}$, where N_p is the number of proposals. To get final proposals set, we need to make score fusion to get final confidence score, then suppress redundant proposals based on these score.

Score fusion for retrieving. To achieve better retrieving performance, for each candidate proposal φ, we fuse its confidence score with its boundary probabilities by multiplication to get the final confidence score p_f:

$$p_f = p_{conf} \cdot p_{t_s}^s \cdot p_{t_e}^e. \tag{4}$$

After score fusion, we can get generated proposals set $\Psi_p = \{\varphi_n = (t_s, t_e, p_f)\}_{n=1}^{N_p}$, where p_f is used for proposals retrieving. In section 4.2, we explore the recall performance with and without confidence score generated by proposal evaluation module.

Redundant proposals suppression. Around a ground truth action instance, we may generate multiple proposals with different temporal overlap. Thus we need to suppress redundant proposals to obtain higher recall with fewer proposals.

Soft-NMS [33] is a recently proposed non-maximum suppression (NMS) algorithm which suppresses redundant results using a score decaying function. First all proposals are sorted by their scores. Then proposal φ_m with maximum score is used for calculating overlap IoU with other proposals, where scores of highly overlapped proposals is decayed. This step is recursively applied to the remaining proposals to generate re-scored proposals set. The Gaussian decaying function of Soft-NMS can be denoted as:

$$p'_{f,i} = \begin{cases} p_{f,i}, & iou(\varphi_m, \varphi_i) < \theta \\ p_{f,i} \cdot e^{-\frac{iou(\varphi_m, \varphi_i)^2}{\varepsilon}}, & iou(\varphi_m, \varphi_i) \geq \theta \end{cases} \tag{5}$$

where ε is parameter of Gaussian function and θ is pre-fixed threshold. After suppression, we get the final proposals set $\Psi'_p = \left\{\varphi_n = (t_s, t_e, p'_f)\right\}_{n=1}^{N_p}$.

4 Experiments

4.1 Dataset and setup

Dataset. ActivityNet-1.3 [1] is a large dataset for general temporal action proposal generation and detection, which contains 19994 videos with 200 action classes annotated and was used in the ActivityNet Challenge 2016 and 2017. ActivityNet-1.3 is divided into training, validation and testing sets by ratio of 2:1:1. **THUMOS14** [2] dataset contains 200 and 213 temporal annotated untrimmed videos with 20 action classes in validation and testing sets separately. The training set of THUMOS14 is the UCF-101 [34], which contains trimmed videos for action recognition task. In this section, we compare our method with state-of-the-art methods on both ActivityNet-1.3 and THUMOS14.

Evaluation metrics. In temporal action proposal generation task, Average Recall (AR) calculated with multiple IoU thresholds is usually used as evaluation metrics. Following conventions, we use IoU thresholds set $[0.5 : 0.05 : 0.95]$ in ActivityNet-1.3 and $[0.5 : 0.05 : 1.0]$ in THUMOS14. To evaluate the relation between recall and proposals number, we evaluate AR with Average Number of proposals (AN) on both datasets, which is denoted as AR@AN. On ActivityNet-1.3, area under the AR vs. AN curve (AUC) is also used as metrics, where AN varies from 0 to 100.

In temporal action detection task, mean Average Precision (mAP) is used as evaluation metric, where Average Precision (AP) is calculated on each action class respectively. On ActivityNet-1.3, mAP with IoU thresholds $\{0.5, 0.75, 0.95\}$ and average mAP with IoU thresholds set $[0.5 : 0.05 : 0.95]$ are used. On THUMOS14, mAP with IoU thresholds $\{0.3, 0.4, 0.5, 0.6, 0.7\}$ is used.

Implementation details. For visual feature encoding, we use the two-stream network [11] with architecture described in [35], where BN-Inception network [36] is used as temporal network and ResNet network [37] is used as spatial network. Two-stream network is implemented using Caffe [38] and pre-trained on ActivityNet-1.3 training set. During feature extraction, the interval σ of snippets is set to 16 on ActivityNet-1.3 and is set to 5 on THUMOS14.

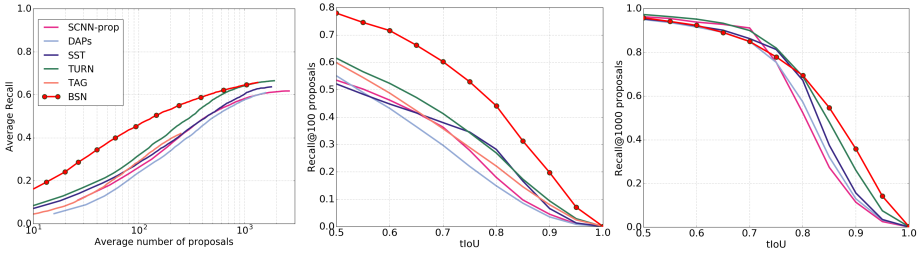


Fig. 4: Comparison of our proposal generation method with other state-of-the-art methods in THUMOS14 dataset. **(left)** BSN can achieve significant performance gains with relatively few proposals. **(center)** Recall with 100 proposals vs tIoU figure shows that with few proposals, BSN gets performance improvements in both low and high tIoU. **(right)** Recall with 1000 proposals vs tIoU figure shows that with large number of proposals, BSN achieves improvements mainly while tIoU > 0.8.

On ActivityNet-1.3, since the duration of videos are limited, we follow [39] to rescale the feature sequence of each video to new length $l_w = 100$ by linear interpolation, and the duration of corresponding annotations to range [0,1]. In BSN, temporal evaluation module and proposal evaluation module are both implemented using Tensorflow [40]. On both datasets, temporal evaluation module is trained with batch size 16 and learning rate 0.001 for 10 epochs, then 0.0001 for another 10 epochs, and proposal evaluation module is trained with batch size 256 and same learning rate. For Soft-NMS, we set the threshold θ to 0.8 on ActivityNet-1.3 and 0.65 on THUMOS14 by empirical validation, while ε in Gaussian function is set to 0.75 on both datasets.

4.2 Temporal Proposal Generation

Taking a video as input, proposal generation method aims to generate temporal proposals where action instances likely to occur. In this section, we compare our method with state-of-the-art methods and make external experiments to verify effectiveness of BSN.

Comparison with state-of-the-art methods. As aforementioned, a good proposal generation method should generate and retrieve proposals to cover ground truth action instances with *high recall* and *high temporal overlap* using relatively few proposals. We evaluate these methods in two aspects.

First we evaluate the ability of our method to generate and retrieve proposals with high recall, which is measured by average recall with different number of proposals (AR@AN) and area under AR-AN curve (AUC). We list the comparison results of ActivityNet-1.3 and THUMOS14 in Table 1 and Table 2 respectively, and plot the average recall against average number of proposals curve of THUMOS14 in Fig 4 (left). On THUMOS14, our method outperforms other state-of-the-art proposal methods when proposal number varies from 10 to 1000. Especially, when average number of proposals is 50, our method significantly improves average recall from 21.86% to 37.46% by 15.60%. On ActivityNet-1.3, our method outperforms other state-of-the-art proposal generation methods on both validation and testing set.

Second, we evaluate the ability of our method to generate and retrieve proposals with high temporal overlap, which is measured by recall of multiple IoU thresholds. We plot the recall against IoU thresholds curve with 100 and 1000 proposals in Fig 4

Table 1: Comparison between our method with other state-of-the-art proposal generation methods on validation set of ActivityNet-1.3 in terms of AR@AN and AUC.

Method	Zhao et al. [24]	Dai et al. [42]	Yao et al. [43]	Lin et al. [39]	BSN
AR@100 (val)	63.52	-	-	73.01	**74.16**
AUC (val)	53.02	59.58	63.12	64.40	**66.17**
AUC (test)	-	61.56	64.18	64.80	**66.26**

Table 2: Comparison between our method with other state-of-the-art proposal generation methods on THUMOS14 in terms of AR@AN.

Feature	Method	@50	@100	@200	@500	@1000
C3D	DAPs [5]	13.56	23.83	33.96	49.29	57.64
C3D	SCNN-prop [7]	17.22	26.17	37.01	51.57	58.20
C3D	SST [3]	19.90	28.36	37.90	51.58	60.27
C3D	TURN [6]	19.63	27.96	38.34	53.52	**60.75**
C3D	BSN + Greedy-NMS	27.19	35.38	43.61	53.77	59.50
C3D	BSN + Soft-NMS	**29.58**	**37.38**	**45.55**	**54.67**	59.48
2-Stream	TAG [24]	18.55	29.00	39.61	-	-
Flow	TURN [6]	21.86	31.89	43.02	57.63	64.17
2-Stream	BSN + Greedy-NMS	35.41	43.55	52.23	**61.35**	**65.10**
2-Stream	BSN + Soft-NMS	**37.46**	**46.06**	**53.21**	60.64	64.52

(center) and (right) separately. Fig 4 (center) suggests that our method achieves significant higher recall than other methods with 100 proposals when IoU threshold varied from 0.5 to 1.0. And Fig 4 (right) suggests that with 1000 proposals, our method obtains the largest recall improvements when IoU threshold is higher than 0.8.

Furthermore, we make some controlled experiments to confirm the contribution of BSN itself in Table 2. For video feature encoding, except for two-stream network, C3D network [12] is also adopted in some works [3,5,6,7]. For NMS method, most previous work adopt Greedy-NMS [41] for redundant proposals suppression. Thus, for fair comparison, we train BSN with feature extracted by C3D network [12] pre-trained on UCF-101 dataset, then perform Greedy-NMS and Soft-NMS on C3D-BSN and original 2Stream-BSN respectively. Results in Table 2 show that (1) C3D-BSN still outperforms other C3D-based methods especially with small proposals number, (2) Soft-NMS only brings small performance promotion than Greedy-NMS, while Greedy-NMS also works well with BSN. These results suggest that the architecture of BSN itself is the main reason for performance promotion rather than input feature and NMS method.

These results suggest the effectiveness of BSN. And BSN achieves the salient performance since it can generate proposals with (1) ***flexible temporal duration*** to cover ground truth action instances with various durations; (2) ***precise temporal boundary*** via learning starting and ending probability using temporal convolutional network, which brings high overlap between generated proposals and ground truth action instances; (3) ***reliable confidence score*** using BSP feature, which retrieves proposals properly so that high recall and high overlap can be achieved using relatively few proposals. Qualitative examples on THUMOS14 and ActivityNet-1.3 datasets are shown in Fig 5.

Generalizability of proposals. Another key property of a proposal generation method is the ability to generate proposals for unseen action classes. To evaluate this property, we choose two semantically different action subsets on ActivityNet-1.3: “Sports, Ex-

Table 3: Generalization evaluation of BSN on ActivityNet-1.3. *Seen* subset: "Sports, Exercise, and Recreation"; *Unseen* subset: "Socializing, Relaxing, and Leisure".

	Seen (validation)		*Unseen* (validation)	
	AR@100	AUC	AR@100	AUC
BSN trained with *Seen* + *Unseen* (training)	72.40	63.80	71.84	63.99
BSN trained with *Seen* (training)	72.42	64.02	**71.32**	**63.38**

ercise, and Recreation" and "Socializing, Relaxing, and Leisure" as *seen* and *unseen* subsets separately. *Seen* subset contains 87 action classes with 4455 training and 2198 validation videos, and *unseen* subset contains 38 action classes with 1903 training and 896 validation videos. To guarantee the experiment effectiveness, instead of two-stream network, here we adopt C3D network [44] trained on Sports-1M dataset [45] for video features encoding. Using C3D feature, we train BSN with *seen* and *seen+unseen* videos on training set separately, then evaluate both models on *seen* and *unseen* validation videos separately. As shown in Table 3, there is only slight performance drop in unseen classes, which demonstrates that BSN has great generalizability and can learn a generic concept of temporal action proposal even in semantically different unseen actions.

Effectiveness of modules in BSN. To evaluate the effectiveness of temporal evaluation module (TEM) and proposal evaluation module (PEM) in BSN, we demonstrate experiment results of BSN with and without PEM in Table 4, where TEM is used in both results. These results show that: (1) using only TEM without PEM, BSN can also reach considerable recall performance over state-of-the-art methods; (2) PEM can bring considerable further performance promotion in BSN. These observations suggest that TEM and PEM are both effective and indispensable in BSN.

Boundary-Sensitive Proposal feature. BSP feature is used in proposal evaluation module to evaluate the confidence scores of proposals. In Table 4, we also make ablation studies of the contribution of each component in BSP. These results suggest that although BSP feature constructed from boundary regions contributes less improvements than center region, best recall performance is achieved while PEM is trained with BSP constructed from both boundary and center region.

4.3 Action Detection with Our Proposals

To further evaluate the quality of proposals generated by BSN, we put BSN proposals into "detection by classifying proposals" temporal action detection framework with state-of-the-art action classifier, where temporal boundaries of detection results are provided by our proposals. On ActivityNet-1.3, we use top-2 video-level class generated by classification model [46][1] for all proposals in a video. On THUMOS14, we use top-2 video-level classes generated by UntrimmedNet [48] for proposals generated by BSN and other methods.Following previous works, on THUMOS14, we also implement SCNN-classifier on BSN proposals for proposal-level classification and adopt Greedy NMS as [7]. We use 100 and 200 proposals per video on ActivityNet-1.3 and THUMOS14 datasets separately.

[1] Previously, we adopted classification results from result files of [47]. Recently we found that the classification accuracy of these results are unexpected high. Thus we replace it with classification results of [46] and updated all related experiments accordingly.

Table 4: Study of effectiveness of modules in BSN and contribution of components in BSP feature on THUMOS14, where PEM is trained with BSP feature constructed by *Boundary* region (f_s^A, f_e^A) and *Center* region (f_c^A) independently and jointly.

	Boundary	*Center*	@50	@100	@200	@500	@1000
BSN without PEM			30.72	40.52	48.63	57.78	63.04
BSN with PEM	✓		35.61	44.86	52.46	60.00	64.17
		✓	36.80	45.65	52.63	60.18	64.22
	✓	✓	**37.46**	**46.06**	**53.21**	**60.64**	**64.52**

Table 5: Action detection results on validation and testing set of ActivityNet-1.3 in terms of mAP@$tIoU$ and average mAP, where our proposals are combined with video-level classification results generated by [46].

	validation				testing
Method	0.5	0.75	0.95	Average	Average
Wang et al. [47]	42.28	3.76	0.05	14.85	14.62
SCC [49]	40.00	17.90	4.70	21.70	19.30
CDC [50]	43.83	25.88	0.21	22.77	22.90
TCN [42]	-	-	-	-	23.58
SSN [51]	39.12	23.48	5.49	23.98	28.28
Lin et al. [39]	44.39	29.65	7.09	29.17	32.26
BSN + [46]	**46.45**	**29.96**	**8.02**	**30.03**	**32.84**

Table 6: Action detection results on testing set of THUMOS14 in terms of mAP@$tIoU$, where classification results generated by UntrimmedNet [48] and SCNN-classifier [7] are combined with proposals generated by BSN and other methods.

Action Detection Methods						
Detection Method		0.7	0.6	0.5	0.4	0.3
SCNN [7]		5.3	10.3	19.0	28.7	36.3
SMS [30]		-	-	17.8	27.8	36.5
CDC [50]		8.8	14.3	24.7	30.7	41.3
SSAD [25]		7.7	15.3	24.6	35.0	43.0
TCN [42]		9.0	15.9	25.6	33.3	-
R-C3D [52]		9.3	19.1	28.9	35.6	44.8
SS-TAD [26]		9.6	-	29.2	-	45.7
SSN [51]		-	-	29.1	40.8	50.6
CBR [32]		9.9	19.1	31.0	41.3	50.1
Proposal Generation Methods + Action Classifier						
Proposal method	Classifier	0.7	0.6	0.5	0.4	0.3
SST [3]	SCNN-cls	-	-	23.0	-	-
TURN [6]	SCNN-cls	7.7	14.6	25.6	33.2	44.1
SST [3]	UNet	4.7	10.9	20.0	31.5	41.2
TURN [6]	UNet	6.3	14.1	24.5	35.3	46.3
BSN	SCNN-cls	15.0	22.4	29.4	36.6	43.1
BSN	UNet	**20.0**	**28.4**	**36.9**	**45.0**	**53.5**

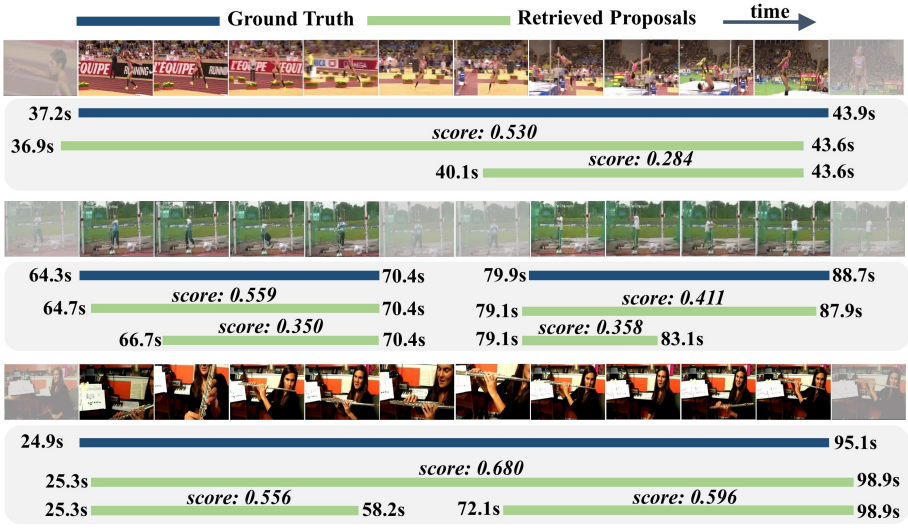


Fig. 5: Qualitative examples of proposals generated by BSN on THUMOS14 (top and middle) and ActivityNet-1.3 (bottom), where proposals are retrieved using post-processed confidence score.

The comparison results of ActivityNet-1.3 shown in Table 5 suggest that detection framework based on our proposals outperforms other state-of-the-art methods. The comparison results of THUMOS14 shown in Table 6 suggest that (1) using same action classifier, our method achieves significantly better performance than other proposal generation methods; (2) comparing with proposal-level classifier [7], video-level classifier [48] achieves better performance on BSN proposals and worse performance on [3] and [6] proposals, which indicates that confidence scores generated by BSN are more reliable than scores generated by proposal-level classifier, and are reliable enough for retrieving detection results in action detection task; (3) detection framework based on our proposals significantly outperforms state-of-the-art action detection methods, especially when the overlap threshold is high. These results confirm that proposals generated by BSN have high quality and work generally well in detection frameworks.

5 Conclusion

In this paper, we have introduced the Boundary-Sensitive Network (BSN) for temporal action proposal generation. Our method can generate proposals with flexible durations and precise boundaries via directly combing locations with high boundary probabilities, and make accurate retrieving via evaluating proposal confidence score with proposal-level features. Thus BSN can achieve high recall and high temporal overlap with relatively few proposals. In experiments, we demonstrate that BSN significantly outperforms other state-of-the-art proposal generation methods on both THUMOS14 and ActivityNet-1.3 datasets. And BSN can significantly improve the detection performance when used as the proposal stage of a full detection framework.

References

1. Caba Heilbron, F., Escorcia, V., Ghanem, B., Carlos Niebles, J.: Activitynet: A large-scale video benchmark for human activity understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2015) 961–970
2. Jiang, Y.G., Liu, J., Zamir, A.R., Toderici, G., Laptev, I., Shah, M., Sukthankar, R.: Thumos challenge: Action recognition with a large number of classes. In: ECCV Workshop. (2014)
3. Buch, S., Escorcia, V., Shen, C., Ghanem, B., Niebles, J.C.: SST: Single-stream temporal action proposals. In: IEEE International Conference on Computer Vision. (2017)
4. Caba Heilbron, F., Carlos Niebles, J., Ghanem, B.: Fast temporal activity proposals for efficient detection of human actions in untrimmed videos. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 1914–1923
5. Escorcia, V., Heilbron, F.C., Niebles, J.C., Ghanem, B.: Daps: Deep action proposals for action understanding. In: European Conference on Computer Vision, Springer (2016) 768–784
6. Gao, J., Yang, Z., Sun, C., Chen, K., Nevatia, R.: Turn tap: Temporal unit regression network for temporal action proposals. arXiv preprint arXiv:1703.06189 (2017)
7. Shou, Z., Wang, D., Chang, S.F.: Temporal action localization in untrimmed videos via multi-stage cnns. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 1049–1058
8. Wang, H., Kläser, A., Schmid, C., Liu, C.L.: Action recognition by dense trajectories. In: Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, IEEE (2011) 3169–3176
9. Wang, H., Schmid, C.: Action recognition with improved trajectories. In: Proceedings of the IEEE International Conference on Computer Vision. (2013) 3551–3558
10. Feichtenhofer, C., Pinz, A., Zisserman, A.: Convolutional two-stream network fusion for video action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 1933–1941
11. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: Advances in Neural Information Processing Systems. (2014) 568–576
12. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 4489–4497
13. Wang, L., Xiong, Y., Wang, Z., Qiao, Y.: Towards good practices for very deep two-stream convnets. arXiv preprint arXiv:1507.02159 (2015)
14. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2014) 580–587
15. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 1440–1448
16. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in neural information processing systems. (2015) 91–99
17. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE transactions on pattern analysis and machine intelligence **32**(9) (2010) 1627–1645
18. Uijlings, J.R., van de Sande, K.E., Gevers, T., Smeulders, A.W.: Selective search for object recognition. International journal of computer vision **104**(2) (2013) 154–171
19. Zitnick, C.L., Dollár, P.: Edge boxes: Locating object proposals from edges. In: European Conference on Computer Vision, Springer (2014) 391–405

20. Kuo, W., Hariharan, B., Malik, J.: Deepbox: Learning objectness with convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 2479–2487
21. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. arXiv preprint arXiv:1612.03144 (2016)
22. Gidaris, S., Komodakis, N.: Locnet: Improving localization accuracy for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 789–798
23. Singh, G., Cuzzolin, F.: Untrimmed video classification for activity detection: submission to activitynet challenge. arXiv preprint arXiv:1607.01979 (2016)
24. Zhao, Y., Xiong, Y., Wang, L., Wu, Z., Lin, D., Tang, X.: Temporal action detection with structured segment networks. arXiv preprint arXiv:1704.06228 (2017)
25. Lin, T., Zhao, X., Shou, Z.: Single shot temporal action detection. In: Proceedings of the 25nd ACM international conference on Multimedia. (2017)
26. Buch, S., Escorcia, V., Ghanem, B., Fei-Fei, L., Niebles, J.C.: End-to-end, single-stream temporal action detection in untrimmed videos. In: Proceedings of the British Machine Vision Conference. (2017)
27. Karaman, S., Seidenari, L., Del Bimbo, A.: Fast saliency based pooling of fisher encoded dense trajectories. In: ECCV THUMOS Workshop. (2014)
28. Oneata, D., Verbeek, J., Schmid, C.: The lear submission at thumos 2014. ECCV THUMOS Workshop (2014)
29. Wang, L., Qiao, Y., Tang, X.: Action recognition and detection by combining motion and appearance features. THUMOS14 Action Recognition Challenge **1** (2014) 2
30. Yuan, Z., Stroud, J.C., Lu, T., Deng, J.: Temporal action localization by structured maximal sums. arXiv preprint arXiv:1704.04671 (2017)
31. Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Van Gool, L.: Temporal segment networks: towards good practices for deep action recognition. In: European Conference on Computer Vision, Springer (2016) 20–36
32. Gao, J., Yang, Z., Nevatia, R.: Cascaded boundary regression for temporal action detection. arXiv preprint arXiv:1705.01180 (2017)
33. Bodla, N., Singh, B., Chellappa, R., Davis, .L.S.: Improving object detection with one line of code. arXiv preprint arXiv:1704.04503 (2017)
34. Soomro, K., Zamir, A.R., Shah, M.: Ucf101: A dataset of 101 human actions classes from videos in the wild. arXiv preprint arXiv:1212.0402 (2012)
35. Xiong, Y., Wang, L., Wang, Z., Zhang, B., Song, H., Li, W., Lin, D., Qiao, Y., Gool, L.V., Tang, X.: Cuhk & ethz & siat submission to activitynet challenge 2016. arXiv preprint arXiv:1608.00797 (2016)
36. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning. (2015) 448–456
37. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2016) 770–778
38. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. In: Proceedings of the 22nd ACM international conference on Multimedia, ACM (2014) 675–678
39. Lin, T., Zhao, X., Shou, Z.: Temporal convolution based action proposal: Submission to activitynet 2017. arXiv preprint arXiv:1707.06750 (2017)
40. Abadi, M., Agarwal, A., Barham, P., et al.: Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467 (2016)
41. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: IEEE Conference on Computer Vision and Pattern Recognition. (2005) 886–893

42. Dai, X., Singh, B., Zhang, G., Davis, L.S., Chen, Y.Q.: Temporal context network for activity localization in videos. In: 2017 IEEE International Conference on Computer Vision (ICCV), IEEE (2017) 5727–5736
43. Ghanem, B., Niebles, J.C., Snoek, C., Heilbron, F.C., Alwassel, H., Khrisna, R., Escorcia, V., Hata, K., Buch, S.: Activitynet challenge 2017 summary. arXiv preprint arXiv:1710.08011 (2017)
44. Tran, D., Ray, J., Shou, Z., Chang, S.F., Paluri, M.: Convnet architecture search for spatiotemporal feature learning. arXiv preprint arXiv:1708.05038 (2017)
45. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. (2014) 1725–1732
46. Zhao, Y., Zhang, B., Wu, Z., Yang, S., Zhou, L., Yan, S., Wang, L., Xiong, Y., Lin, D., Qiao, Y., Tang, X.: Cuhk & ethz & siat submission to activitynet challenge 2017. arXiv preprint arXiv:1710.08011 (2017)
47. Wang, R., Tao, D.: Uts at activitynet 2016. AcitivityNet Large Scale Activity Recognition Challenge **2016** (2016) 8
48. Wang, L., Xiong, Y., Lin, D., Van Gool, L.: Untrimmednets for weakly supervised action recognition and detection. arXiv preprint arXiv:1703.03329 (2017)
49. Heilbron, F.C., Barrios, W., Escorcia, V., Ghanem, B.: Scc: Semantic context cascade for efficient action detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Volume 2. (2017)
50. Shou, Z., Chan, J., Zareian, A., Miyazawa, K., Chang, S.F.: Cdc: Convolutional-deconvolutional networks for precise temporal action localization in untrimmed videos. arXiv preprint arXiv:1703.01515 (2017)
51. Xiong, Y., Zhao, Y., Wang, L., Lin, D., Tang, X.: A pursuit of temporal accuracy in general activity detection. arXiv preprint arXiv:1703.02716 (2017)
52. Xu, H., Das, A., Saenko, K.: R-c3d: Region convolutional 3d network for temporal activity detection. arXiv preprint arXiv:1703.07814 (2017)